REMARKS

Claims 1-4, and 19 stands rejected under 35 USC §102(e) as being anticipated by Chao et al, U.S. patent 6,081,507. Claims 5-8 and 12-18 stand rejected under 35 USC §103 as being unpatentable over Chao et al, U.S. patent 6,081,507 in view of Duckering et al., U.S. patent 6,721,325. Claims 9-11 and 20 stands rejected under 35 USC §103 as being unpatentable over Chao et al, U.S. patent 6,081,507 in view of Tayyar et al., U.S. Pub. No. 2003/0050945 and in view of Duckering et al., U.S. patent 6,721,325.

Claims 1, 5-9, 12 and 19 have been amended to more clearly state the invention. Reconsideration and allowance of each of claims 1-20, as amended, is respectfully requested.

Chao et al, U.S. patent 6,081,507 discloses methods and apparatus for handling time stamp aging. Compensating for time stamp aging in systems employing fair packet queuing algorithms by (i) representing a time stamp of each head-of-line packet of a session with a finite number of bits, (ii) representing a system potential with a finite number of bits, (iii) storing a packet's time stamp when it is served, (iv) storing an obsolete indicator for each stored time stamp, and (v) updating the obsolete indicator or purging obsolete time stamps.

Chao at column 1, lines 15-22 states: In general, the present invention concerns congestion control and traffic management in networks and inter-networks operating at relatively high data rates and carrying information which may have differing quality of service (or "QoS") requirements. In particular, the present invention concerns methods and apparatus for fairly servicing queues at an output port of a switch (for switching ATM packets for example) or router (for routing TCP/IP packets for example). Stated at column 13, lines 28-42: Finally, the present invention provides techniques for

addressing a time stamp aging problem. In any scheduler, when an k^{th} packet of session i is served (i.e., transmitted), the time stamp F_i^{k-1} may be stored in a look-up table for later use (as F_i^{k-1}). The look-up table can be placed in memory for supporting a large number (N) of sessions (or flows), with the entry of F_i^{k-1} addressed by i (where i=0,1, . . . ,N-1). Besides the time stamp F_i^{k-1} other information related to session (or flow) i can also be stored at (or pointed to from) the same location. Later, when a new packet k of the session (or flow) i arrives at the head of the session queue, and thus becomes the head-of-line (or "HOL") packet, the stored time stamp F_i^{k-1} is needed so that it may be compared with the system potential $v(a_i^k)$ for determining a new starting potential $S_i^{k-1}$for the k k^{th} packet as discussed above.

Chao at column 18, lines 55-67 and column 19, lines 1-19 states: In accordance with the present invention, a previous time stamp F_i^{k-1} may be considered to be obsolete if the system potential $v(a_i^k)$ exceeds it. That is, once the system potential $v(a_i^k)$ is larger than F_i^{k-1}, it will remain so. (Naturally, updating will occur when the next packet of the i.sup.th session or flow is served.) In the present invention, a number of bits can be used to record (i) a number of overflow events of the system potential $v(a_i^k)$, and (ii) a time zone where the system potential $v(a_i^k)$ and the stored finish potential F_i^{k-1}, respectively, belong. A purging means may be used to purge all stored time stamps F_i^{k-1} that have become obsolete. The purging means should run fast enough to check each of the stored time stamps and purge all obsolete ones before the history of the system potential $v(a_i^k)$ overflows due to its representation by a finite number of bits.

Each purging operation has one, and perhaps two, memory accesses. The first is to read the time stamp F_i^{k-1} of the last departed packet. If that time stamp F_i^{k-1} is obsolete (i.e., less than the current system potential $v(a_i^k)$), the second memory access is a write operation to mark the time stamp as obsolete. Due to the limited speed of memory accesses, it might not be possible to complete all purging operations during a time slot, particularly when N is large. Since it might not be possible to perform all N purging operations during a time slot (i.e., it might take a number of time slots to perform all N purging operations), the present invention may track any time stamp or system potential overflow while all purging operations are performed. For example, in the present invention, a first counter variable $C_v(t)$ may be used to track system potential overflow, while another counter variable C_i may be used to track time stamp (or virtual finish time) overflow.

Chao at column 22, lines 47-67 and column 23, lines 1-20 states: FIG. 17 is a high level flow diagram of an exemplary method 1700 for scheduling the service of head-of-line packets in a number of flow queues. First, as shown in step 1710, a search is performed to find the flow queue with a head-of-line packet having a lowest time stamp. (Recall from FIG. 14 that a memory having storage locations based on the time stamps, stores corresponding flow queue identifiers.) An exemplary method for performing this step is described with reference to FIG. 18 below. Next, as shown in decision step 1720, it is determined whether any more head-of-line packets have the same time stamp. (Recall from FIG. 14 that there may be a linked list of flow queue identifiers pointed to from a storage location corresponding to a given time stamp.) If

not, the validity bit is reset (to "0") as shown in step 1730. Further, since bits in strings at higher levels in the hierarchy may be affected by the change of the validity bit, these bits are also reset, if necessary. An exemplary method for performing this step is described with reference to FIG. 20. Processing then continues to decision step 1740.

Returning to decision step 1720, if there are more packets with the time stamp, then the validity bit should remain "1" and therefore, no changes are needed. Thus, in this case, processing may continue directly to decision step 1740. At decision step 1740, it is determined whether there is a new head-of-line packet in a flow queue. This will occur in a flow queue, having more than one packet, in which the head-of-line packet is serviced. This will also occur if a previously empty flow queue receives a packet(s). In any event, if a flow queue has a new head-of-line packet processing branches to step 1750 where the identification of the flow queue is pointed to from the address defined by the time stamp of the new head-of-line packet. Next, as shown in step 1760, the validity bit, as well as bits in other strings higher in the hierarchy, may be updated if necessary. An exemplary method for performing steps 1750 and 1760 is described with reference to FIG. 19. Processing then continues to step 1710. Returning to step 1740, if a new head-of-line packet is not in a flow queue (which will only happen in the event that all flow queues are empty), processing continues directly to step 1710.

Chao at column 36, lines 53-67 and column 37, lines 1-12; As mentioned above, to ensure unambiguous comparisons between the system potential v(t) and each stored time stamp F in any of the T time slots, the counter variable C_v (t) should be able to record at least T+1 times of overflow. In this example, the counter variable C_v (t) is $\log_2 (T+1)=\log_2 (21007)$, which is rounded up to 15 bits. The counter variable C_v (t) is incremented by 1 each time the system potential v(t) overflows. To facilitate purging, an "obsolete" bit O_i, and another $[\log_2 (T+1)]$ bit counter variable C_i is defined for each entry F_i in the look-up table. Thus, referring to FIG. 33, an exemplary look-up table 3300 may have N records 3310, each record having a field 3312 storing the obsolete bit O_j, a field 3314 having the bit string C_i encoding the number of time stamp overflows, and a field 3316 having a bit string F_i encoding the time stamp of the previously served packet of the i^{th} session (or flow). Accordingly, assuming that both F_i^{k-1} and $v(a_i.^{k-})$ are represented by the same number of bits, v(t) and F can be compared directly if they are both in the same time zone (i.e., if C_v (t)=C_i). Otherwise, again assuming that both $F_i s^{k-1}$ and $v(a_j.i^k)$ are represented by the same number of bits, simply comparing the time zones (i.e., Cv_i (t) and C_i) indicates which of $F_i.^{k-1}$ and $v(a_i.^k)$ is larger. In this example, if there are M=32,768 time stamps, the width of each entry 3310 of the look-up table 3300 will be 31 bits. That is, 1 bit for the obsolete bit O_i 15 bits $(=\log_2 (T+1)=\log_2 21006)$ for the overflow counter C_i, and 15 bits $(=\log_2 M=\log 232,768)$ for the time stamp.

Duckering et al., U.S. patent 6,721,325 discloses an apparatus for scheduling multi-service category ATM cell traffic through contention points in an ATM

network is provided. The service categories have predefined delivery priorities according to quality of service guarantees. To satisfy these priorities while maintaining fair treatment to low priority connections, aging markers are incrementally assigned to queued cells and these markers in combination with priority data are used to determine which connection is serviced next. FIG. 1 demonstrates an implementation of a prior art queue servicing solution using multiple shaping calendars 12 which constitutes a particular form of a traffic shaper. The calendars 12 are prioritized. When more than one calendar 12 contains a connection ready to send a cell only the highest priority calendar gets to transmit. All lower priority calendars have to wait until all calendars with higher priority have nothing ready to send. In FIG. 1 a low priority weighted-fair-queuing (WFQ) scheduler 14 which could be in the form of a calendar, a tag comparator or other WFQ method has been added to provide work-conserving support for low priority connections such as UBR. Calendars could be divided based upon the traffic type (real-time or non-real-time) service category (CBR, rtVBR, nrt-VBR, ABR, UBR) delay requirements (such as CDV and max CTD) requirements or a combination of these such as delay requirements for real-time traffic and service category for non-real-time traffic. Since Quality of Service (QoS) can be related to network node CDV and max CTD requirements, dividing calendars within the node based upon these requirements prioritize traffic based upon QoS. FIG. 2 demonstrates one implementation of exhaustive by age priority servicing. Connections are shaped using multiple calendars 20, where a calendar is a traffic shaping entity which, in effect, defines a succession of time intervals, in each of which are identified queues to be

serviced during that interval and each calendar has a programmable 1-point CDV associated with it. The 1-point CDV is described in ATM Forum's Traffic Management Specification Version 4.0. In the multiple calendar architecture illustrated in FIG. 2 the position of each queue within the calendar is determined by the Generic Cell Rate Algorithm (GCRA) also defined in the aforementioned ATM Forum's Traffic Management Specification. In this example a WFQ calendar 22 is also available to provide work conservation for low priority connections, though no aging is associated with it. A method of scheduling traffic through the network that includes scheduling a connection with both a shaper and a WFQ at the same time, it is possible to provide a minimum rate guarantee with work conservation providing any additionally available bandwidth. The shaper will shape to the minimum rate, while the WFQ provides everything above. This supports rate-based backpressure, where the non-real-time traffic can be throttled back to minimum rates simply by halting service of the WFQ.

Tayyar et al., U.S. Pub. No. 2003/0050954 discloses a scheduler which uses a GPS simulation to determine an order in which to service entities uses a novel dynamic data structure with a sophisticated, but simple, pointer update mechanism. Preferred embodiments of the scheduler perform a fixed amount of work per scheduling event. A scheduling event can be either computing a new virtual finish timestamp upon a new arrival to the scheduler, or determining which entities are to leave the GPS system because their finish timestamp has expired. The scheduler may be used in packet scheduling in a packet handling device, such as a router, scheduling access of software processes to a computer processor or the like. The scheduler may implement

weighted fair queuing (WFQ). A time stamper 18 computes a timestamp for each packet. The timestamp is a function of either a virtual time 20 maintained by time stamper 18 or the timestamp of a previous packet 11 which has been assigned to the same session in GPS queues 22 by classifier 14. Time stamper 18 also maintains GPS queues 22 which contain records of any packets 11 in each session maintained by the system in order of their timestamps. The timestamps indicate the order in which the packets 11 would be transmitted if the packets were fluid packets being transmitted in a GPS system.

The present invention provides a method, a scheduler and a computer program product for implementing Quality-of-Service (QoS) scheduling of a plurality of flows with aging time stamps. The present invention ensures the available bandwidth will not be wasted and that the available bandwidth will be efficiently and fairly allocated. The method, scheduler and computer program product of the present invention permit many network traffic flows to be individually scheduled per their respective negotiated Quality-of-Service (QoS) levels, avoiding some disadvantages of prior art arrangements.

Independent claims 1, 9 and 19 respectively recite a method, a scheduler and a computer program product for implementing Quality-of-Service (QoS) scheduling of a plurality of flows with aging time stamps. As amended, each of the independent claims 1, 9 and 19 recite the steps of or a memory manager, or a queue manager, for sequentially accessing a subset of time stamp data from a time stamp aging memory array; each time stamp data subset containing time stamp data for a subplurality of

flows; performing guaranteed aging processing steps for each flow utilizing said time stamp data subsets to identify and mark invalid calendar next time values, identifying a new frame arrival for an empty flow and accessing time stamp data from a flow queue control block (FQCB) for said flow and said flow time stamp data in said time stamp aging memory array; responsive to said identified new frame arrival for said empty flow, checking a selection indicator of said time stamp aging memory array data to identify said target calendar for attaching said flow; each flow being attached to at least one of a plurality of calendars including a low latency service (LLS)/normal latency service (NLS), a peak bandwidth service (PBS) and a weighted fair queue (WFQ) ring; said LLS, NLS, and PBS calendars being time based; and said weighted fair queue (WFQ) ring being weight based; responsive to said selection indicator value, checking a target calendar next time valid bit of said time stamp aging memory array data for said flow; responsive to said target calendar next time valid bit being on, comparing a target calendar next time from said flow queue control block (FQCB) for said flow with a current time; and responsive to said target calendar next time being less than said current time, turning off said target.calendar next time valid bit to mark said target calendar next time as invalid.

The disclosure of the Chao patent is set forth above in detail, and Applicant respectfully submits that the above limitations, as now expressly recited in independent claims 1, 9, and 19, as amended, are neither disclosed nor suggested by Chao.

There are significant differences between what is disclosed in the Chao

patent and the pending claims 1-20, as amended; and the Examiner is respectfully requested to withdraw the rejection of claims 1-4 and 19 under 35 U.S.C. §102 because it is axiomatic that for prior art to anticipate under §102 it has to meet every element of the claimed invention (Hybritech Inc. v. Monoclonal Antibodies, Inc., 802 F.2d 1367, 1379, 231 USPQ 81, 90 (Fed. Cir. 1986)). Reconsideration and allowance of each of the pending claims 1-20, as amended, is respectfully requested.

The Chao patent discloses scheduling where searching is performed to find the flow queue with a head-of-line packet having a lowest time stamp using a memory having storage locations based on the time stamps, that stores corresponding flow queue identifiers and using a linked list of flow queue identifiers pointed to from a storage location corresponding to a given time stamp to determined whether any more head-of-line packets have the same time stamp. The Chao patent discloses storing a packet's time stamp when it is served, storing an obsolete indicator for each stored time stamp, and updating the obsolete indicator to purge obsolete time stamps. The Chao patent does not disclose, nor suggest storing a set of indicator bits, as taught and claimed by Applicants. The Chao patent does not disclose, nor suggest a memory manager as, as taught and claimed by Applicants. The Chao patent does not disclose, nor suggest identifying a new frame arrival for an empty flow and accessing time stamp data from a flow queue control block (FQCB) for said flow and said flow time stamp data in said time stamp aging memory array; responsive to said identified new frame arrival for said empty flow, checking a selection indicator of said time stamp aging memory array data to identify said target calendar for attaching said flow, as taught and

claimed by Applicants. The Chao patent does not disclose, nor suggest each flow being attached to at least one of a plurality of calendars including a low latency service (LLS)/normal latency service (NLS), a peak bandwidth service (PBS) and a weighted fair queue (WFQ) ring; said LLS, NLS, and PBS calendars being time based; and said weighted fair queue (WFQ) ring being weight based.

Further Applicants respectfully submit that considering the total teachings of Chao, Duckering and Tayyar would not achieve the claimed invention as recited in each of the pending independent claims 1, 9 and 19, as amended. Applicants respectfully submit that one of ordinary skill in the art would have been led to the claimed invention by the reasonable teachings or suggestions found in the prior art, including Chao, Duckering and Tayyar. The teachings of Duckering and Tayyar are set forth above.

Applicants respectfully submit that Duckering and Tayyar add nothing to suggest the claimed method, scheduler and a computer program product for implementing Quality-of-Service (QoS) scheduling of a plurality of flows with aging time stamps as respectively recited in independent claims 1, 9 and 19, as amended. Duckering and Tayyar do not disclose, nor suggest a memory manager and storing a set of indicator bits including a selection indicator, as taught and claimed by Applicants. Thus, each of the independent claims 1, 9 and 19, as amended, is patentable.

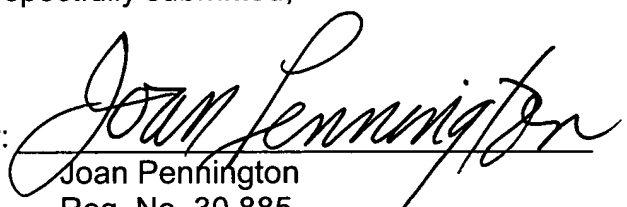
Dependent claims 2-8, 10-18, and 20 respectively depend from patentable claims 1, 9, and 19, further defining the invention. Each of the dependent claims 2-8, 10-18, and 20, as amended, is likewise patentable.

Applicants have reviewed all the art of record, and respectfully submit that the claimed invention is patentable over all the art of record, including the references not relied upon by the Examiner for the rejection of the pending claims.

It is believed that the present application is now in condition for allowance and allowance of each of the pending claims 1-20, as amended, is respectfully requested. Prompt and favorable reconsideration is respectfully requested.

If the Examiner upon considering this amendment should find that a telephone interview would be helpful in expediting allowance of the present application, the Examiner is respectfully urged to call the applicants' attorney at the number listed below.

Respectfully submitted,

By: 

Joan Pennington
Reg. No. 30,885
Telephone: (312) 670-0736